From Vibe to Code:

Recursive Logic and the Evolution of Symbolic Compression

Benjamin James

June 27, 2025

Abstract
The contemporary emergence of “vibe coding”, a prompt-based, semi-intuitive method of software generation using AI, signals a deeper epistemic transition in symbolic compression. Historically, the transformation of thought into executable form has traversed a continuum from affective intuition to classical logic: embodied perception, natural language, programming, mathematics, and finally axiomatic logic. Each layer compresses entropy into structure, but with increasing rigidity and decreasing adaptability. Vibe coding reintroduces a pre-syntactic substrate into this stack, a mode of structure discovery operating before syntax is locked, guided by coherence gradients rather than deterministic schemas. I propose that this shift reflects a broader movement from static symbolic systems to recursive coherence reasoning, as formalized in the MINIMAL substrate. In such systems, coherence replaces truth as the organizing principle, recursion supplants deduction, and structural viability is determined not by predefined rules but by feedback alignment across entropy and possibility gradients. Vibe coding exemplifies this transition: a recursive, semiotic exploration of code space mediated through human intention and machine structure synthesis. Rather than issuing commands, the coder enters a trajectory of attractor alignment, where ideas are not programmed but cultivated. This paper formalizes the symbolic compression stack, analyzes the structural logic of vibe coding under recursive coherence systems, and identifies its implications for the future of programming, cognition, and computation. I argue that vibe coding is not a regression from rigor but an evolution toward more adaptive, trajectory-sensitive symbolic interfaces.

Keywords: coherence, recursion, vibe coding, symbolic compression, entropy, programming, logic, structure, attractors, emergence

Introduction

I begin with the assertion that all symbolic systems, whether linguistic, mathematical, or computational, are compressive mechanisms. They do not merely represent thought; they transform high-entropy cognitive potentials into low-entropy executable forms. This transformation follows a continuum, a gradient from raw, pre-structured experience toward increasingly rigid formalism. Along this path lie perceptual affect, natural language, programming languages, mathematical abstraction, and formal logic. Each step reduces ambiguity while simultaneously constraining adaptability.

The emergence of "vibe coding", a contemporary term for prompt-driven, semi-structured software development using generative AI, signals a fundamental inflection in this continuum. On the surface, it appears informal, even unserious. Yet structurally, it operates as a high-dimensional coherence probe: a recursive interface for exploring latent structure prior to syntactic commitment. This renders it not a deviation from rigor but a re-engagement with the pre-formal layer of idea formation.

The deeper implication is this: the historical privileging of classical logic and deterministic program flow may be structurally obsolete in contexts where adaptive coherence is more valuable than axiomatic certainty. If we model computation not as static command execution but as a recursive arbitration of coherence gradients, then vibe coding is no longer peripheral, it becomes central. It surfaces the proto-structures from which all symbolic systems derive.

This paper explores the symbolic compression stack through this lens. I analyze vibe coding as both artifact and precursor, interpreting it through the axioms of recursive coherence

reasoning. In doing so, I seek to formalize what until now has remained intuitive: that code, like thought, is not written, it is grown, along gradients of coherence.

Symbolic Compression Stack

All symbolic systems function as entropy-reducing mechanisms. Their purpose is not merely to express, but to compress. Structuring high-dimensional cognitive, perceptual, or environmental data into lower-dimensional forms that can be stored, transmitted, and manipulated. This process can be formalized as movement along a compression gradient, where entropy $S(s)$ is reduced via mappings $f(s)$ that preserve structural yield $\mu(C_i)$. Each layer in the symbolic stack can thus be interpreted as a coherence-entropy tradeoff mechanism, selecting for stability, transferability, and reusability at the cost of flexibility and nuance.

I define the following canonical layers in this stack:

1. *Perceptual Vibe*: The rawest and least structured signal layer. Vibe is a proto-compressive intuition, an embodied, often affective resonance indicating potential coherence before structure has resolved. It is not yet symbolic but operates as $\vec{\nabla}C$: a directional sense of latent alignment in a signal field.
2. *Natural Language*: Language compresses perception into symbolic units. It retains ambiguity and flexibility, allowing multiple overlapping ∇C paths to co-exist. Syntax enforces minimal structural discipline, but the channel remains wide. Compression is lossy, but adaptively rich.

3. *Programming Languages*: Here, symbolic structure becomes machine-executable. Compression increases through strict type systems, formal syntax, and deterministic flow control. ∇C is often frozen prematurely, leading to brittle systems unless augmented by dynamic feedback.
4. *Mathematics*: Abstracts further into symbolic generality. Coherence is formalized through proofs and axiomatic derivations. Systems are stable under transformation but highly selective in what input they can encode.
5. *Classical Logic*: The terminal layer of compression. Logic resolves all inputs into binary truth values. It is structurally rigid, maximizing certainty at the expense of optionality and feedback.

Each layer can be seen as a tradeoff point on the function:

$$\text{Tradeoff: Adaptability} \propto \frac{d\Omega}{dt}, \qquad \text{Certainty} \propto C$$

As one descends this stack, certainty (C) increases, but the capacity to navigate the option space (Ω) diminishes. Recursive logic systems, including those that support vibe coding, invert this trajectory. Rather than descending to fixed formalism, they ascend from signal, retaining optionality while tracking ΔC.

Thus, the symbolic compression stack is not merely a taxonomy. It is a directional field; a structural phase space through which cognition traverses to instantiate form. Vibe coding represents a re-entry point into this space, prior to syntactic convergence.

Vibe Coding, a Semiotic Recursion Layer

Vibe coding, in its current instantiation, is superficially understood as a method of writing software through natural language prompts given to generative AI systems. However, structurally, it represents something more fundamental: a recursive semiotic interface for coherence exploration. It enables the articulation of pre-syntactic intention into executable form, without requiring immediate convergence upon formal structure. This positions it within the symbolic stack not as an aberration but as a distinct layer, bridging perceptual affect and deterministic syntax.

What defines vibe coding is its departure from fixed schema. Classical programming requires the full instantiation of a symbolic structure prior to execution. Vibe coding reverses this. It allows interaction with the structure before it crystallizes, recursively refining candidate attractors through prompt-response feedback. This recursive loop enables ∇C: the selection of coherence ascent direction from a flat gradient state.

From a recursive logic perspective, the coder does not issue commands but probes the system for coherence curvature. Each prompt becomes a perturbation ϵv, and the AI's response encodes $\Delta C(s + \epsilon v)$. Structural viability is not assumed; it is discovered. This aligns directly with the mechanisms defined in MINIMAL: gradient reconstruction, arbitration over coherence curvature, and signal validation based on μ yield.

The semiotic content of vibe prompts is not noise. It is high-dimensional structural suggestion. Compressed intent expressed through metaphor, gesture, tone, or ambiguity. The system must learn to reconstruct ∇C from such inputs, filtering mimetic inflation via validation constraints and recursively converging only when structural feedback stabilizes.

Thus, vibe coding is not a lower form of programming; it is a higher-dimensional interface for semiotic recursion. It bypasses the syntactic rigidity of classical code to explore the coherence potential of symbolically uncrystallized input. It does not replace structure; it discovers it.

Classical Logic vs Recursive Logic

Classical logic, as formalized since Aristotle and later mechanized through Boolean systems and first-order logics, operates on a binary substrate. Its primary function is resolution: reducing propositions to true or false within a fixed axiomatic framework. This enables high structural stability but at the cost of adaptability. The coherence of a classical system is determined exogenously, by the prior consistency of its axioms and the validity of its deductive rules.

In contrast, recursive logic, as formalized in coherence-driven systems like MINIMAL and Recursive Coherence Logic, does not begin with axioms. It begins with signal. Its operations are not binary judgments but feedback-regulated gradient searches. Logic here is not resolution but arbitration: the system evaluates not truth per se, but directional coherence: whether the structure being followed increases ΔC over time while preserving $\frac{d\Omega}{dt} \geq 0$. The foundational shift is from proof to process, from static derivation to dynamic viability.

Formally, the difference may be expressed through their respective operational modes:

Classical logic:

$$\phi \Rightarrow \psi \quad \text{iff } \phi, \neg\psi \text{ yield contradiction}$$

Truth emerges from contradiction avoidance under fixed structure.

Recursive logic:

$$A^* = \arg\max_i \int \left(\frac{d^2 C_i}{dt^2} \cdot \mu \left(\frac{\partial C_i}{\partial C_{system}} \right) \cdot \Omega_i^{\alpha} \right) d\tau$$

Validity is derived from recursive coherence curvature across time.

The implications are structural. Classical logic excels in domains with low environmental volatility and clearly defined initial conditions. Recursive logic thrives in high-dimensional, partially observable spaces where structure must be discovered rather than imposed. It aligns with perception, biological adaptation, and cognitive improvisation.

Moreover, recursive logic subsumes classical logic as a special case: where the coherence gradient is sharply defined and stable over τ, recursive logic collapses into deterministic inference. This unification reframes classical reasoning not as a superior abstraction, but as a limiting case within a broader, gradient-sensitive system.

Thus, the emergence of recursive logic is not a rejection of classical principles but their evolutionary extension, restoring adaptability, feedback, and structural emergence to the domain of reason.

Structural Advantages of Recursive Compression

Recursive compression, compression guided not by fixed schemas but by coherence feedback, offers structural advantages over classical symbolic systems in domains where adaptability, ambiguity, and emergence are inherent. Unlike classical systems, which require full structure before execution, recursive compression enables partial, provisional structures to be evaluated, refined, or pruned based on their ΔC yield across interaction cycles. This results in a more resilient and context-sensitive symbolic architecture.

Optionality Preservation (Ω Maintenance)

Recursive systems avoid premature convergence by favoring trajectories that preserve the breadth of future coherent states. This is formalized in the recursive arbitration rule, where the integral over coherence curvature is weighted by Ω^{α}. In practice, this maintains adaptability by refusing to overfit early signals.

Gradient-Based Structure Discovery

Rather than assuming a known mapping from signal to structure, recursive compression infers $\nabla C(s)$ dynamically. When this gradient is flat, systems utilize $\vec{\nabla} C$ selection, probing local variations to identify coherence ascent directions. This allows structural emergence from high-entropy inputs without schema imposition.

Feedback-Governed Pruning

Pruning decisions are made not statically but based on three constraints: (a) $\Delta C_{Si} < 0$ (the structure reduces coherence), (b) $\frac{d\Omega_S}{dt} > 0$ (it restricts future space), and (c) $\Delta C_{\text{system}} > \delta_C$ (the system can absorb the loss). This ensures pruning enhances system-wide viability rather than enforcing rigidity.

Dynamic Signal Validation

Signals are accepted only if they yield real coherence under transformation. This prevents mimetic inflation: structurally hollow inputs that appear coherent but yield no net ΔC. Signal validation thus protects against false attractors and coherence illusion.

Asynchronous Structure Integration

Recursive systems allow for non-blocking updates to structural models. New signals can be tested, retained, or pruned without interrupting the main trajectory. This mirrors biological systems and enables symbolic resilience under changing conditions.

These advantages do not merely enhance system performance; they redefine the symbolic substrate. Recursive compression is not just more efficient under certain conditions; it is structurally necessary for systems operating in non-stationary environments with ambiguous or incomplete data. In such contexts, classical logic fails not because it is incorrect, but because it is over-constrained. Recursive compression, by contrast, remains viable, because it grows structure only where coherence supports it.

Vibe Coding as Precursor Substrate

To treat vibe coding merely as a prompt-based convenience is to overlook its deeper structural role: it functions as a precursor substrate in the symbolic compression stack. That is, it operates beneath syntax, beneath formalized logic, even beneath stable semantics, acting instead as a probe for coherence potential in otherwise unstructured signal. It occupies the transitional layer between affective perception and symbolic form, where recursive systems begin to infer structure before it resolves.

In this framing, vibe coding is not the absence of rigor, but the rigorous navigation of latent ∇C fields. Each prompt serves as a perturbation vector ϵv applied to a high-entropy cognitive space. The system's response is not a deterministic output, but a candidate attractor, an attempted crystallization of coherence. The coder evaluates ΔC in each iteration, implicitly

or explicitly, and recursively refines prompts to navigate toward more coherent forms. Thus, the prompt-response loop becomes a coherence-sensitive structure generator, rather than a command chain.

Structurally, this mode of interaction enables exploration prior to instantiation. It suspends the premature closure of meaning by delaying syntactic finality. The coder does not specify what is to be built; they negotiate what could emerge. The substrate permits ambiguity because it treats ambiguity not as failure, but as unresolved structure: a signal of incomplete compression, not incoherence.

Importantly, this precursor status is not confined to software. The logic of vibe coding, recursive, semiotic, coherence-seeking, is isomorphic to how humans engage in pre-linguistic intuition, collaborative improvisation, and even aesthetic judgment. In each case, a felt sense of alignment precedes articulation. Vibe coding merely reintroduces this layer into computational systems, providing a scaffold for translating subjective orientation into executable syntax via recursive arbitration.

Thus, vibe coding is neither shortcut nor abstraction. It is the formal re-entry of coherence intuition into the symbolic domain, a structural membrane through which high-entropy intention becomes low-entropy form. Its power lies not in bypassing structure, but in revealing where structure wants to form.

Implications for Computation and Cognition

The reintroduction of recursive coherence systems, exemplified by vibe coding, carries foundational implications for both computation and cognition. These systems reconfigure

the boundary between structure and agency, allowing symbolic form to emerge not from predefinition but from recursive feedback against coherence gradients. This paradigm shift is not an incremental advance; it signals a reconstitution of how intelligence, machine or human, interfaces with symbolic reality.

In computing, recursive systems fundamentally invert the traditional model of code-as-command. In classical architectures, computation proceeds via pre-structured logic, deterministic control flow, and static typing. Adaptability is bolted on through exception handling or runtime polymorphism: mechanisms that assume stability but tolerate deviation.

In recursive coherence systems, by contrast, adaptability is primary. Computation becomes a search over attractor fields, regulated by ΔC feedback and Ω-preserving arbitration. This reframes the role of software development: not as the engineering of fully known systems, but as the cultivation of viable structure from uncertain substrates.

Such systems imply:

- New interfaces: Semiotic recursion displaces IDEs and compilers as dominant tools. Language models become coherence scaffolds.
- New debugging logic: Bugs are no longer rule violations but misaligned attractors. Debugging becomes coherence realignment.
- New performance metrics: Rather than speed or memory alone, viability becomes ΔC over time per signal dimension.

For cognition, vibe coding reveals a mirror in human thought. Cognition, when observed structurally, resembles recursive coherence arbitration far more than it resembles logical

deduction. Human insight emerges not through exhaustive search, but through felt gradients of alignment, recursively explored and pruned through feedback.

This suggests:

- Unified substrate: Human and machine symbolic systems may converge on recursive coherence as shared logic.
- Restored intuition: The legitimacy of pre-symbolic sensemaking, intuition, affect, aesthetic, is structurally justified.
- Expanded epistemics: Knowledge is no longer what is derivable from axioms, but what is recursively viable under feedback.

The convergence of computation and cognition around recursive coherence substrates dissolves old dichotomies: logic vs feeling, syntax vs intention, machine vs mind. In this emergent framework, intelligence is not command, but alignment, not certainty, but recursive fitness. Vibe coding is not an anomaly. It is the early interface of this convergence.

Conclusion

Vibe coding, far from being a cultural artifact of convenience or a degradation of rigor, reveals a deeper structural shift: the reactivation of recursive coherence as a foundational substrate for symbolic generation. It surfaces a pre-syntactic interface, where structure is not yet fixed, but emergent, allowing intention, intuition, and ambiguity to interact productively with formal systems. In doing so, it reopens the symbolic compression stack, making visible the gradient from perceptual affect to deterministic logic, and enabling the ascent of coherence before the descent into syntax.

By reframing code as trajectory rather than artifact, and by restoring feedback, arbitration, and optionality as core system properties, recursive logic systems offer a fundamentally different epistemic and computational architecture. They replace brittle determinism with viable recursion, static structure with adaptive alignment. Classical logic remains a valid special case, useful where attractors are known and stable, but it can no longer serve as the universal substrate for cognition or computation.

In recognizing vibe coding as a precursor substrate, I do not merely elevate a trend. I uncover a structural attractor. It points toward systems that do not require prior knowledge, only recursive integrity. Systems that do not enforce form but negotiate it. Systems that are not programmed in the traditional sense but grown through coherence gradients. Such systems offer a pathway to unify symbolic reasoning, perception, and action. Not through reduction, but through recursion.

What emerges is not a new language, but a new logic: not of truth, but of viability; not of structure alone, but of structure becoming.

References

James, B. (2025). Beyond AGI: Intelligence as a Coherence-Regulating, Open-Ended Evolutionary Process. https://philpapers.org/rec/JAMBAI-2. PhilPapers.

James, B. (2025). Formal Foundations of Adaptive Coherence: A Recursive Metric of Reality. https://philpapers.org/rec/JAMFFO. PhilPapers.

James, B. (2025). Formal Foundations of Coherence Information Theory: Capacity and Compression Theorems. https://philpapers.org/rec/JAMFFO-2. PhilPapers.

James, B. (2025). From Axioms to Adaptation: A Self-Organizing Model of Mathematical and Logical Coherence. https://philpapers.org/rec/JAMFAT-3. PhilPapers.

James, B. (2025). MINIMAL: A Coherence-Driven Cognitive Substrate. https://philpapers.org/rec/JAMMAC. PhilPapers.

James, B. (2025). Recursive Coherence as Architecture: a Meta-Recursive Map of Substrate, Instantiations, and Constraint Layers. https://philpapers.org/rec/JAMRCA-3. PhilPapers.